Available online at www.sciencedirect.com

SciVerse ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Efficient authentication for fast handover in wireless mesh networks

Celia Li, Uyen Trang Nguyen, Hoang Lan Nguyen, Nurul Huda*

Department of Computer Science & Engineering, York University, 4700 Keele Street, Toronto, Ontario, Canada M3J 1P3

ARTICLE INFO

Article history:
Received 21 September 2012
Received in revised form
3 June 2013
Accepted 10 June 2013

Keywords:
Wireless mesh networks
Network security
Authentication
Mobile devices
Hand-over
Hand-off
Key management
Privacy

ABSTRACT

We propose new authentication protocols to support fast handover in IEEE 802.11-based wireless mesh networks. The authentication server does not need to be involved in the handover authentication process. Instead, mesh access points directly authenticate mobile clients using tickets, avoiding multi-hop wireless communications in order to minimize the authentication delay. Numerical analysis and simulation results show that the proposed handover authentication protocol significantly outperforms IEEE 802.11 authentication in terms of authentication delay.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

A wireless mesh network (WMN) consists of mesh clients and mesh points (routers). Mesh clients can be static (e.g., desktops, database servers) or mobile hosts (e.g., cell phone, laptops, PDAs). The MPs form a wireless mesh backbone to provide multi-hop connectivity from one mesh client to another or to the Internet. A subset of mesh points act as *mesh access points* (MAPs), connecting mesh clients to the WMN. A small number of mesh points work as gateways, connecting the WMN to the Internet.

A WMN is dynamically self-organized and self-configured, with nodes in the network automatically establishing and maintaining mesh connectivity among themselves. This feature brings many benefits to IEEE 802.11-based mesh networks such as low installation cost, large-scale deployment, fault-tolerance, and self-management.

Wireless mesh networks support many important applications such as Internet access provisioning in rural areas, ad hoc networking for emergency and disaster recovery, security surveillance, and information services in public transportation systems, airports, shopping malls, and stadiums. The technology enables networking capability where wiring or installing cables is difficult or expensive and deployment time is a concern.

With the rapid growth of mobile services for handheld devices such as smartphones, tablets and laptops, Internet connectivity anytime anywhere has become a necessity in every day life, business, education and entertainment. While cellular networks effectively handle the handoff problem

* *Corresponding author.*
E-mail addresses: cli@cse.yorku.ca (C. Li), utn@cse.yorku.ca (U.T. Nguyen), lan@cse.yorku.ca (H.L. Nguyen), huda@cse.yorku.ca (N. Huda).

0167-4048/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cose.2013.06.001

using signaling embedded in their low-level protocols, there are currently no efficient, transparent handoff solutions for IEEE 802.11-based wireless networks. At the moment, these networks, even if they give the appearance of continuous connectivity to a roaming client, provide connections that are in fact often interrupted when a client transfers from one access point to the next, because handover delays can be as long as several seconds (Velayos and Karlsson, 2003). For some applications (e.g. transferring files), this delay is acceptable; however, it is far too long for real-time traffic such as interactive voice over IP or video conferencing (Amir and Danilov, 2006).

The current version of wireless mesh networking standards IEEE 802.11s does not specify any mechanisms to support fast hand-off for mobile clients. A mesh client has to be authenticated by an authentication server via *multi-hop wireless communications*, which may result in long delay, low reliability and thus potential service interruption. A performance study of message transmission delay in IEEE 802.11-based mesh networks by Srivatsa and Xie (2008) shows that as the number of wireless hops between two routers increases from one to five, the delay of a message between a client and an authentication server increases from 0.15 s to 0.8 s. Since the authentication process involves several messages (e.g., nine messages in the EAP-TLS protocol used by 802.11s), the handoff latency may be several seconds long, which is not tolerable in real-time applications such as VoIP, newscast, and stock quote distribution.

Our work in this paper contributes toward extending the IEEE 802.11s standards to support fast roaming for mobile clients. In particular, we focus on fast authentication during the hand-off process as well as during the initial login time. We propose a new trust model for WMNs based upon which our proposed authentication protocols are designed. We present ticket-based authentication protocols that are efficient and resilient to attacks. The authentication server does not need to be involved in the handover authentication. Instead, mobile clients' authentications are done by mesh access points, avoiding multi-hop wireless communications. Fast authentication from one MAP to another during the hand-off process is supported using tickets (Kohl and Neuman, 1993). Numerical analysis and simulation results show that our login authentication protocol improves the latency of 802.11s login authentication, and our handover authentication protocol supports fast authentication during the hand-off process, which is lacking in 802.11s.

The remainder of the paper is organized as follows. Related work is discussed in Section 2. We describe the ticket types used in the proposed authentication protocols in Section 3. In Section 4, we present our login and handover authentication protocols. Security analysis is discussed in Section 5. Performance evaluations of the proposed protocols are given in Section 6. Section 7 concludes the paper and outlines our future work.

2. Related work

We first identify the requirements of an authentication protocol designed specifically for WMNs.

- The protocol must incur low computation costs due to mobile devices' limited computational capabilities, storage and/or power supply. The number of messages to be exchanged should be minimized due to the low bandwidth of wireless channels (compared with wired networks).
- The delay of re-authentication during the hand-off process should be low to avoid service interruption.
- The protocol must support mutual authentication between a client and a MAP, protection of client identity privacy, and resilient to various types of attacks (Horn et al., 2002) such as forgery, replay attack, denial of service attack, time-memory trade-off attack, and identity privacy attack. (These types of attacks will be defined and discussed in Section 5.)
- The amount of control traffic generated by mobility management mechanisms, such as handover authentication, has a significant impact on the overall network performance. Network operators are interested in reducing the amount of control traffic in their networks (possibly at the expense of higher server loads or lower handover performance (Kassab, 2007)).

We broadly divide authentication protocols for *wireless networks* in standards and in literature into three categories: multi-hop authentication, pro-active authentication, and ticket-based authentication. In multi-hop authentication protocols (IEEE, 2003; Forsberg et al., 2008; Jiang et al., 2006; Buddhikot, 2003; Shi, 2007), when a mobile client moves from one access point (network) to another, it has to be re-authenticated by the authentication server (home network) which may be located many hops away from the client. *Multi-hop wireless* communications incur long latency and may lead to service interruptions. Pro-active authentication protocols (Mishra et al., 2004; Park et al., 2007) attempt to minimize the authentication latency during the handover process by distributing pairwise master keys (PMK), proofs of successful log-in authentications, to potential target access points of a mobile client before the client moves to another access point. Ticket-based authentication protocols (Kassab, 2007; Li, 2010) also try to minimize the authentication latency during the handover process by using tickets as proofs of successful log-in authentications.

Pro-active and ticket-based authentication protocols follow the principles of single sign-on. They both execute login authentication one time and then calculate a PMK shared by a mobile client and a nearby access point. With the knowledge of the PMK, a client can be connected to that access point and authenticated quickly in the future.

2.1. *Multi-hop authentication*

The current wireless mesh networking standard IEEE 802.11s (IEEE, 2009; Hiertz, 2010) uses IEEE 802.11i security standards (IEEE, 2003). Using IEEE 802.11i login authentication protocol, such as EAP-TLS, a client is authenticated by an authentication server (AS), which may be many hops away from the client. When the client transfers from one MAP to another, he/she has to be re-authenticated by the AS, which incurs long latency.

IEEE 802.11F or Inter-Access Point Protocol (IAPP) is an optional extension to IEEE 802.11 that provides wireless access

point communications among multi-vendor systems. When a client moves away from its current access point, it may start to search for a new access point (AP). If a new AP is located, the client will send a re-association request frame to the new AP. The request contains the client's MAC address and the basic service set identifier (BSSID) of the old AP. Upon receiving the re-association request frame, the new AP sends an access-request message to the authentication server (AS) to verify the BSSID of the old AP. If that BSSID is valid, the AS will send an access-accept message to the new AP which contains security information for handoff communications between the old and new APs. IAPP supports secure exchanges of clients' security information (e.g., cryptographic keys) between the current AP and a new AP during the handoff process. However, IAPP does not effectively reduce the handoff latency because both the current and new APs have to communicate with a Remote Authentication Dial In User Service (RADIUS) server during the handover process (Huang, 2006; Kassab, 2005).

Also in this category is the Protocol for Carrying Authentication for Network Access (PANA) (Forsberg et al., 2008), a network-layer transport for the Extensible Authentication Protocol (EAP) defined in IEEE 802.11 standards that enables authentication between clients and access networks. PANA runs between a client and a server in order to perform authentication and authorization for the network access service. PANA does not define any new authentication mechanisms, but carries the EAP payload instead, which performs authentication. Therefore, authentication during the handover still has to be performed via the multi-hop wireless communication mechanism of EAP.

In mobile IP, virtual home environment (3GPP Technical Specification 22.121 v5.3.1, 2002; Marenic, 2003) and cellular networks, the foreign agent/network must communicate with a client's home agent/network via multi-hop communications to authenticate the client (Jiang et al., 2006; Buddhikot, 2003; Shi, 2007). The SIM card of a client and the authentication center of the client's home network are pre-installed with a shared secret key K. When the client roams to a foreign network, the foreign agent must communicate with the client's home network in order to obtain the shared key K, which will then be used to authenticate the client. This approach, if applied to WMNs, means multi-hop wireless communications between a home network and every foreign agent, and thus potential service interruption as discussed earlier.

Our proposed authentication protocols rely on one-hop communication during the handover authentication process to minimize the latency and service interruption.

2.2. *Pro-active authentication*

In the handover authentication protocol of IEEE 802.11i standard, after the authentication server successfully authenticates a mobile client, it will send a key called pairwise master key (PMK) to the AP associated with the client. The client will perform the same calculation as the AS to obtain the same PMK. The AP and client will use the PMK to derive a pairwise transient key (PTK) for encrypting future packets exchanged between them (IEEE, 2003). The AS then sends the PMK to the neighbors of the current AP, one by one. The PMK serves as proof of the client's successful log-in authentication performed by the AS. By letting the AS pre-distribute the PMK to the neighbors of the current AP, the client will not need to be authenticated by the AS when it moves to another AP. However, the pre-distribution of keys by the AS incurs extra traffic overhead within the backhaul network. In addition, if the distance between the AS and a neighbor AP is long, the PMK may not arrive in time at the neighbor AP before the client moves and connects to that neighbor AP, causing service interruption. Our proposed handover authentication also uses pre-distribution of keys, but it requires only one local transmission (one broadcast) from the current AP to its neighbors as opposed to multi-hop key pre-distributions in other protocols (Mishra et al., 2004; Park et al., 2007) which impose extra traffic overhead in the backhaul network.

Consider the mesh network shown in Fig. 1. Client C is authenticated successfully and connected to MAP M. The AS then sends a PMK to the neighbors of M, namely MAPs N, R and P. (When client C re-associates with a neighbor in the future, the MAP will use the PMK to authenticate C.) The AS distributes the PMK to N, R and P via three, four and five hops, respectively. This incurs traffic in the backhaul network. Furthermore, if the traffic load in the network is heavy, it may take longer for the PMK to reach the neighbor MAPs, increasing the chance of service interruption if the client is moving fast. In our proposed handover authentication protocol (HAP), the key pre-distribution is between the current MAP and its neighbors, only one hop away. The HAP thus do not impose traffic overhead in the backhaul network, is not sensitive to the traffic load in the backhaul, and minimizes key pre-distribution latency via one-hop communications.

The scheme by Mishra et al. (2004) pre-distributes PMKs using neighbor graphs. Once the mobile station A completes an initial full EAP-TLS authentication with an access point AP_i, the authentication server (AS) determines the neighbors of AP_i using the neighbor graph. The AS then sends a PMK to each neighbor N of node AP_i. (The client A also receives all the PMKs the AS sends to AP_i's neighbors.) If A requests to re-associate with N in the future, N will use the PMK to authenticate A. When A roams and connects itself to a new AP, say AP_j, the AS will in turn distribute a PMK to each of AP_j's neighbors.

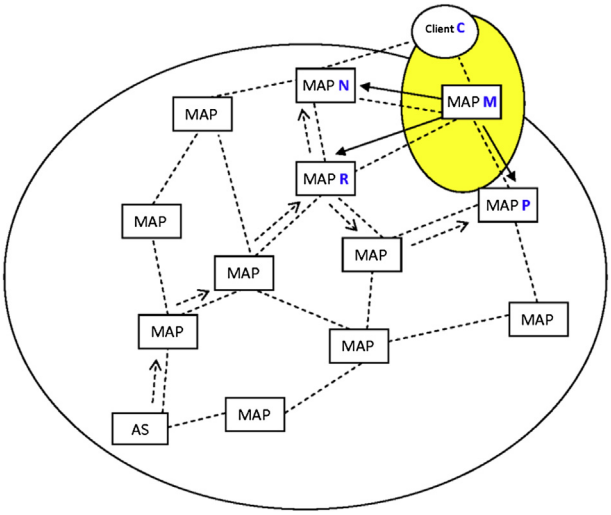


Fig. 1 – Backhaul overhead.

There are two major problems with this scheme. First, as the number of mobile clients in the network increases and as they move around the network, the PMK distribution task is a burden on the AS. Second, the PMK distribution consumes network bandwidth on a global scale.

Park et al. (2007) also use neighbor graphs for pro-active key distribution. However, the AS does not distribute the PMKs. Instead, the AS distributes a set of matrices, which are then used by APs and mobile clients in combination with the key generation process proposed by Du et al. (2003) to generate PMKs. The protocol works as follows. After the AS successfully authenticates a client C using EAP-TLS, it generates two matrices for C: a matrix M of size $h \times N$, where N is the number of APs in the network and $h < N$ is a number chosen by the AS, and a matrix A of size $N \times h$. Let i and j denote the identification numbers of client C and the associated AP, respectively. The AS then sends row $A(i)$ of matrix A and column $M(i)$ of matrix M to client C, and row $A(j)$ of matrix A and a column $M(j)$ of matrix M to the AP. (The matrix information is sent encrypted using the private key shared by the client (or the AP) and the AS.) The client and its associated AP then exchange columns $M(i)$ and $M(j)$, which serve as proofs of their initial successful authentications. Next they compute $K_{ij} = A(i) \times M(j)$ and $K_{ji} = A(j) \times M(i)$, respectively. K_{ij} and K_{ji} have the same value (because matrix K is symmetric), which is the PMK shared by the client and the AP. This scheme suffers from the same drawbacks as the algorithm by Mishra et al. (2004).

Our proposed handover authentication distributes a shared key between neighboring mesh access points (i.e., local traffic) and do not involve the authentication server. This minimizes global traffic overheads and key pre-distribution latency.

2.3. Ticket-based authentication

Li (2010) proposes a ticket-based authentication protocol to support fast handover in wireless local area networks (WLAN). It is a pro-active key distribution approach. After the AS successfully authenticates a mobile client C, it sends a set of tickets to C, one for each neighbor AP of the AP C is currently connected to. A ticket for a neighbor N contains the (encrypted) PMK to be shared by C and N later if C will move to the service area of N. The AS also distributes the PMKs stored in the set of tickets C owns to the neighbor APs in preparation for C's roaming. The major drawback of this scheme is the distributions of PMKs to the neighbor APs, which is acceptable in the wired backbone of a WLAN, but bandwidth-consuming in the wireless backbone of a WMN. In addition, the AS has to generate a large number of tickets, one for each client–AP pair, in the network. (In our proposed protocols, the AS issues only one ticket per client.)

The protocol by Kassab (2007) is very similar to that by Mishra et al. (2004) discussed earlier. After the AS successfully authenticates a mobile client C, it sends a set of PMKs to C and the neighbor APs of the AP C is currently associated with, one PMK for a client–AP pair. When C roams to a neighbor AP N, it generates a ticket that is encrypted with the PMK shared by C and N. N will use the shared PMK to verify the ticket and authenticate C. This protocol has the same disadvantages as the pro-active key distribution scheme by Mishra et al. (2004).

Shames Qazi (2008) proposes a ticket-based authentication scheme for wireless mesh networks. The authentication server assigns tickets to registered mesh clients so that they can communicate with each other. The scheme is designed for authentications between mesh clients, and not between mesh access points and clients. In addition, it does not provide any solutions for fast authentication during handoff.

Anmin Fu (2010) proposes a fast handover authentication mechanism based on tickets for IEEE 802.16m (mobile WiMAX). In this scheme, all the access points and clients of a network are considered as a group and share a group key. After the AS successfully authenticates a client C, it generates a ticket for C, which is encrypted with the group key, and sends the ticket to C. When C moves to another access point N, it submits the ticket to N, which will verify the ticket using the group key. In large mesh networks, using a single group key for the whole network is not a secure nor scalable method.

A qualitative comparison of our proposed login authentication protocol (LAP) and handover authentication protocol (HAP) with other protocols is given in Table 1, where n denotes the number of neighbor MAPs of the MAP to which a client is currently connected.

The objective of our proposed authentication protocols is to support fast authentication during the login time and the hand-off process in a secure, scalable manner with low overhead. The major difference between our handover authentication scheme and the other ticket-based protocols is that in ours keys and tickets needed for a handover are distributed by a MAP to its one-hop neighbors, while in the other protocols they are distributed by the authentication server which are typically multiple hops away from the neighbor MAPs. The proposed protocols are thus suitable for real-time applications in WMNs. They are built upon a new trust model (Li and Nguyen, 2010) and different types of tickets described next.

3. Proposed trust model and ticket types

We present the definition of ticket and the trust model upon which our authentication protocols are built. We also describe in detail the different types of tickets used in the proposed authentication protocols. Refer to Table 2 for the notation used in the remainder of the article.

3.1. Ticket overview

Our proposed trust model is based on the concept of ticket from Kerberos and a Kerberos-assisted authentication scheme proposed by Pizada and McDonald for mobile ad-hoc networks (Pizada and McDonald, 2004). A ticket serves as a pass that a user submits to a system/network to allow it to verify the user's identity. One Kerberos ticket can be used for multiple services in the same system/network. Within the lifetime of a ticket, only a one-time authentication using password is required. As a result, tickets offer better security, more convenience and faster authentication then traditional authentication schemes using passwords (Jablon, 2001).

Kerberos, however, is a centralized authentication scheme and not suitable for use in WMNs where distributed operations are desirable. For example, a Kerberos ticket is bound to the

Table 1 – Comparison of authentication approaches.

Protocol	Type of authentication	Login or handover?	# of Hops[a]	AS involved?	Backhaul overhead[b]	Neighbor graph required?
EAP-TLS (IEEE, 2009; Hiertz, 2010)	Multi-hop	Login	Multiple	Yes	9	No
IAPP (Huang, 2006; Kassab, 2005)	Multi-hop	Handover	Multiple	Yes	2	No
PANA (Forsberg et al., 2008)	Multi-hop	Login	Multiple	Yes	7	No
Mobile IP (Jiang et al., 2006; Buddhikot, 2003; Shi, 2007)	Multi-hop	Handover	Multiple	Yes	2	No
LAP	Multi-hop	Login	One	No	0	No
802.11i handover (IEEE, 2003)	Pro-active	Handover	Multiple	Yes	n	No
Mishra et al. (2004)	Pro-active	Handover	Multiple	Yes	max $3n$	Yes
Park et al. (2007)	Pro-active	Handover	Multiple	Yes	$2n+1$	Yes
Kassab (2007)	Ticket-based	Handover	Multiple	Yes	$n+1$	Yes
Anmin Fu (2010)	Ticket-based	Handover	Multiple	Yes	n	No
HAP	Ticket	Handover	One	No	0	No

a Number of hops between a client and the authenticator.
b Number of messages exchanged between MAPs and the AS to prepare for a handover.

network that issues the ticket. A client must present its ticket to each network it visits. The home authentication server has to be involved for verifying the ticket and authenticating the client. In wireless multi-hop routing environments such as inter-domain mesh networks, the communication between the client in a foreign network and the home authentication server may incur unacceptable delay and service interruption while the client roams among networks. Our proposed trust model, ticket design and authentication protocols aim at minimizing the latency of the handover authentication process and service interruption.

3.2. *The proposed trust model*

The proposed trust model (shown in Fig. 2) is built upon the concept of "ticket" and "ticket agent". In this paper, A ticket is used to establish the trust relationships among entities.

Table 2 – Notation.

Notation	Description
C	Client
R	Mesh access point (MAP)
A	Ticket agent
I_x	ID number of entity x
Θ_C	Transfer ticket issued to a client
P_x	Public key issued to x
T_x	Ticket issued to x
τ_{exp}	Expiry date and time of a ticket
N_x	A nonce generated by x
Sig_x	Digital signature of entity x
MAC_{alg}	Type of MAC algorithm
$E_{pub_x}(m)$	Encryption of message m using x's public key
$D_{pub_x}(m)$	Decryption of message m using x's public key
$E_K(m)$	Encryption of message m using a shared key K
$D_K(m)$	Decryption of message m using a shared key K
$E_{K_{MAC}}(m)$	Encryption of message m using MAC key K_{MAC}
K_{MAC}	The key used to produce a message authentication code (Section 3.3.3)
$V_k(m)$	Message authentication code (MAC) resulting from the application of a MAC algorithm and a MAC key k on a message m

A ticket agent is a trusted third party who issues and manages various types of tickets and can be trusted by various entities in a mesh network. A ticket agent's role can be compared to public-key certificate authorities or credit card issuers.

Following are the trust relationships among the network entities shown in Fig. 2:

- Ticket agent–mesh access points (MAPs): The mutual trust between a MAP and its ticket agent is established via the public key certificates issued by a certificate authority (CA). The trust is established when a MAP applies for a MAP ticket from a ticket agent.
- Ticket agent–client: The mutual trust is based on the public key certificates issued by the CA and is established when a client applies for a client ticket from a ticket agent.
- MAP–client: The mutual trust relationship between a client and its home MAP is established via their respective client ticket and MAP ticket, which are described in Sections 3.3.1 and 3.3.2.
- MAP–MAP: Any two neighboring MAPs trust each other via their public key certificates. This trust allows a client to roam among different MAPs in a mesh network.

Obtaining a client ticket or a MAP ticket is done offline before a client joins a network, and not part of authentication

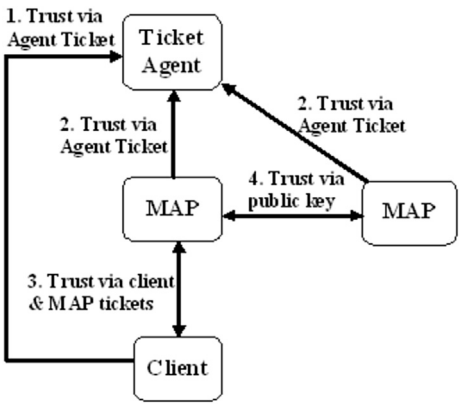


Fig. 2 – Trust model of WMNs.

process. Thus, the public key operations for obtaining tickets do not affect the efficiency of our authentication protocols presented in Section 4.

3.3. Tickets in the authentication protocols

Tickets are issued and managed by ticket agents who are trusted by mesh clients and mesh points to perform such tasks. There can be several ticket agents serving a network. Tickets are used to establish the trust between a ticket agent and a MAP, between a ticket agent and a client, between a MAP and a client, between a MAP and another MAP (see Fig. 2). The lifetime of a ticket is determined by its issuer's policy.

Three types of tickets are used in our authentication protocols: client ticket, MAP ticket and transfer ticket. They are needed for mutual authentication between a client and a MAP when the client logs in to the network, or roams to another MAP.

We will use the notation listed in Table 2 throughout the paper to facilitate the discussions.

3.3.1. Client tickets

A client applies for a client ticket from a ticket agent. The trust between a client and a ticket agent is established through their public key certificates issued by a central authority.

Following is the structure of a client ticket:

$$T_C = \{I_C, I_A, \tau_{exp}, P_C, Sig_A\}$$

- T_C: client ticket issued by ticket agent A whose ID is I_A.
- I_C: ID number of the client that is given this ticket.
- I_A: ID number of the ticket agent who issued the ticket T_C.
- τ_{exp}: expiry date and time of ticket T_C. The ticket agent will re-issue a new ticket for the client if the ticket is expired.
- P_C: public key of client I_C, which is used by a MAP to verify the signature signed by the client in the login authentication protocol (see Section 4.1). The ticket agent obtains the public key from the client's public key certificate. We assume that the agent is a trusted party and has access to public key certificates of all clients and MAPs.
- Sig_A: digital signature of ticket agent I_A, which gives a recipient reason to believe that the ticket was created by ticket agent I_A, and that it was not altered in anyway.

3.3.2. MAP tickets

The operator of a mesh network applies for MAP tickets, one per MAP, and distributes them to the MAPs in the network. The operator is also responsible for requesting and distributing a new MAP ticket before the current MAP ticket expires. Following is the structure of a MAP ticket:

$$T_R = \{I_R, I_A, \tau_{exp}, P_R, Sig_A\}$$

- T_R: MAP ticket issued by ticket agent A whose ID is I_A.
- I_R: ID number of the MAP that is given this ticket.
- I_A: ID number of the ticket agent who issued ticket T_R to MAP R.
- τ_{exp}: expiry date and time of ticket T_R. The ticket agent will re-issue a new ticket for the MAP once the current ticket expires.
- P_R: public key of MAP I_R, which will be used by clients to verify the signature of MAP R in messages R sends. The ticket agent obtains the public key from the MAP's public key certificate.
- Sig_A: digital signature of ticket agent I_A.

3.3.3. Transfer tickets

A transfer ticket is used to establish the trust relationship between a MAP and a client when a client roams from one MAP to another. When a client C first logs in to the network, it sends its client ticket to a nearby MAP M_1, which will authenticate the client. If the authentication succeeds, M_1 will issue to C a transfer ticket and become the home MAP of C. (We borrow the terminology from mobile IP.) When C roams to a foreign MAP M_2, it submits the transfer ticket to M_2 for authentication. The transfer ticket proves to the foreign MAP that client C has been successfully authenticated by its home MAP.

The structure of a transfer ticket Θ_C is as follows:

$$\Theta_C = \{\mu, V_{K_{MAC}}(\mu)\}, \text{ where } \mu = \{I_R, I_C, I_A, \tau_{exp}, MAC_{alg}\}$$

Message μ stores the information of the client, home MAP and ticket agent as follows:

- I_R: ID number of the MAP who issues this transfer ticket.
- I_C: ID number of the client who owns this transfer ticket.
- I_A: ID number of the ticket agent who issued C's client ticket.
- τ_{exp}: expiry date and time of this ticket.
- MAC_{alg}: message authentication code algorithm. (The inclusion of the type of MAC algorithm in a transfer ticket is optional. It is not required if the parties agree on an algorithm in advance.)

We now discuss about the value $V_{K_{MAC}}(\mu)$ stored in the transfer ticket and the use of the MAC algorithm. During the authentication between client C and its home MAP M_1, they exchange two partial keys (also called *nonces*[1]) N_{C1} and N_{R1} (see Section 4.1 for details of the authentication procedure). They will both then compute a shared key $K_{MAC} = N_{C1}||N_{R1}$, where $||$ denotes a concatenation. M_1 subsequently applies the MAC algorithm and key K_{MAC} to message μ to produce a MAC value $V_{K_{MAC}}(\mu)$, which will protect message μ, and thus the transfer ticket, against forgery and unauthorized modifications. M_1 combines message μ and $V_{K_{MAC}}(\mu)$ to form the transfer ticket to be sent to C.

When client C moves into contact with a foreign MAP (e.g. M_2), to prepare for a handover to the new MAP, C submits the transfer ticket issued by M_1 to the foreign MAP (e.g. M_2) for authentication (step (3) in Fig. 3).

In order to allow a foreign MAP (e.g. M_2) to process the transfer ticket and authenticate C, the home MAP M_1 is required to securely send the key $K_{MAC} = N_{C1}||N_{R1}$ to the foreign MAP (e.g. M_2) in advance. (We describe in Section 4.2 how to deliver key K_{MAC} from the home MAP to any of its neighbor in a timely, secure and efficient manner.)

[1] Such a partial key is used only once and cannot be re-used by the party that created it in the first place. In this article, we call these partial keys *nonces* to simplify the presentation.

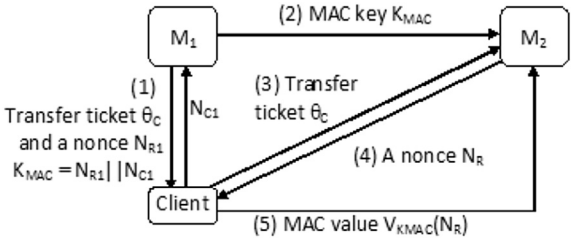


Fig. 3 – Information exchange between a client and MAPs.

The foreign MAP (e.g. M_2) will use key K_{MAC} and the MAC algorithm to verify the authenticity and data integrity of the transfer ticket Θ_C submitted by client C. (M_2 will also verify the identity of C in the handover authentication protocol described in Section 4.2, and illustrated by steps (4) and (5) in Fig. 3.)

It should be noted that each ticket has its own expiration date. The synchronization of ticket updates follows the timing synchronization function (TSF) of the 802.11s standard (Wang and Lim, 2008). The lifetime of a key K_{MAC} is the same as that of the transfer ticket associated with it. A foreign MAP in the network can re-issue a new transfer ticket for the ticket owner if the current transfer ticket is about to expire.

Readers may note that the formats of the above tickets are similar to that of X.509 certificates. However, our tickets contain extra information that cannot be accommodated by X.509 format, e.g., client ID in a client ticket, MAP ID in a MAP ticket, and the MAC value in a transfer ticket.

4. The proposed authentication protocols

We propose two authentication protocols, one for the initial login into a network and the other for subsequent roaming (handover). Our authentication protocols follow a key hierarchical structure similar to that in IEEE 802.11i (IEEE, 2003). That is, a pairwise master key (PMK) is created during the authentication process, and a pairwise transient key (PTK) and a group transient key (GTK) are derived from the PMK subsequently. The two parties involved in the authentication will used the PTK for point-to-point communications and the GTK for group communications (broadcast, multicast) between them.

Public key operations are computationally intensive. Mobile devices, on the other hand, have limited computing capability and power resources. Therefore, our design of the proposed authentication protocols aims to minimize

- the number of message exchanges between a mobile client and MAPs or the authentication server, thus minimizing the authentication latency and resource consumption by the mobile device;
- the number of public key operations performed by mobile devices, thus minimizing resource consumptions by mobile devices.

In addition, we aim to minimize the number of multi-hop communications, thus minimizing the authentication latency and traffic in the backhaul network. At the same time, we ensure that the protocols are secure and scalable. Note that MAPs are not computationally constrained and typically have constant power supplies; thus we are not concerned about them regarding public key operations.

4.1. The login authentication protocol (LAP)

The trust between a client and a MAP is established via the client ticket and the MAP ticket. Since an agent is a *trusted* authority, a client ticket (or a MAP ticket) issued in advance by the agent is the proof of the authentication between the agent and the corresponding client (or MAP).

Following are the order of the messages to be exchanged in the protocol and explanation:

$$
\begin{array}{l}
(1)\ C \longrightarrow R\text{: } I_C \\
(2)\ R \longrightarrow C\text{: } T_R \\
(3)\ C \longrightarrow R\text{: } E_{P_R}(M_C), \text{ where } M_C = \{T_C, N_{C1}, N_{C2}\} \\
(4)\ R \longrightarrow C\text{: } E_{P_C}(M_R), \text{ where } M_R = \{N_{R1}, N_{R2}\} \\
(5)\ C \longrightarrow R\text{: } N_{R2} \\
(6)\ R \longrightarrow C\text{: } N_{C2}, \Theta_C
\end{array}
$$

(1) A client C requests to join a network and associate with a MAP. C sends a request message containing its ID number to the MAP.

(2) A MAP R replies with a message which contains its MAP ticket to inform mesh clients and neighboring MAPs of its presence and ID. Client C verifies the digital signature of the ticket agent A who issued the MAP ticket T_R using A's public key. (We assume that client C and MAP R have the public key certificate of the ticket agent.) C also verifies other information in the MAP ticket such as the ID of the ticket agent and the ticket expiry date.

(3) If the above verifications are successful, C extracts the MAP's public key from the MAP ticket T_R (see Section 3.3.2) and generates a message M_C which contains C's client ticket T_C and two nonces N_{C1} and N_{C2}. C then encrypts the message using the MAP's public key $(E_{P_R}(M_C))$ and sends the encrypted message to the MAP R. Upon receiving the message, R decrypts it using its private key, and verifies the digital signature of the ticket agent who issued the client ticket T_C (using the ticket agent's public key). R then verifies other information recorded in the client ticket T_C such as the ID of the ticket agent who issued T_C and the ticket expiry date.

(4) If the above verifications succeed, MAP R retrieves the client's public key from ticket T_C (see Section 3.3.1), and generates a message M_C containing two nonces N_{R1} and N_{R2}. R then encrypts message M_C using the client's public key $(E_{P_C}(M_R))$, and sends the encrypted message to client C. C will decrypt the message using its private key to N_{R1} and N_{R2}. Both the client and the MAP then calculate their shared MAC key $K_{MAC} = N_{C1}||N_{R1}$, where the operator $||$ denotes a concatenation, and N_{C1} and N_{R1} are the nonces generated in steps (2) and (3) above. (The security of nonces N_{C1} and N_{R1}, and thus key K_{MAC}, is ensured by the MAP's and client's public–private keys.)

(5) Client C then sends N_{R2} to the MAP R. Upon receiving this message, the MAP R has successfully authenticated the client C, because only C has the knowledge of N_{R2}.

(6) To allow the client to authenticate the MAP, R sends N_{C2} (generated by C in step (2)) to client C. The MAP also creates a transfer ticket Θ_C for C, and subsequently sends a message containing both the N_{C2} and the transfer ticket to C. After client C receives N_{C2} correctly, it is considered to have successfully authenticated the MAP because only R has the knowledge of N_{C2}. C will use the transfer ticket Θ_C to roam from one MAP to another in the network.

Following are additional discussions of the above protocol.

(a) Although other clients could see and may attempt to use the transfer ticket, only the rightful owner of the ticket will be able to use it to pass the handover authentication procedure. The ticket has to be used in conjunction with the key K_{MAC}, which only the client owning of the transfer ticket knows (see Section 4.2).
(b) We recommend SHA-2 hash functions for use in the hash-based MAC algorithm because they are employed in several widely-used security applications and protocols. SHA-2 is considered collision resistant (Manuel, 2011).

If the size of the MAC output is L bits, the size k of the MAC key K_{MAC} should be longer than $L/2$ bits. Key sizes of less than $L/2$ bits would decrease the security strength of the function. Keys longer than L bits are acceptable but the extra length would not significantly increase the function strength (RFC 2104 – HMAC: Keyed-Hashing for Message Authentication). Therefore, we recommend a key size of 160 bits, the size of the SHA-2 outputs. As a result, the size of the nonces N_{C1} and N_{R1} (and of the other nonces) is 80 bits.

(c) Key management between a MAP and a client allows them to derive a shared key to be used after the authentication for secure data exchanges. We follow the framework of key management defined in IEEE 802.11i security standards (IEEE, 2003). That is, right after step (4) of the authentication procedure, both parties compute a shared pairwise master key as follows:

$$PMK_0 = N_{C2} \| N_{R1} \quad (1)$$

After the login authentication is completed, the two parties use the pairwise master key PMK_0 to compute a shared key called pairwise transient key (PTK) as specified by IEEE 802.11i security standards (see Section 4.3). The PTK will be used to encrypt packets exchanged between the client and the MAP. When the client moves to a new MAP, the two parties will compute a new set of pairwise master key (PMK) and PTK to be shared between themselves. The computations of the new PMK and PTK are discussed in Sections 4.2 and 4.3, respectively.

4.2. *Handover authentication protocol (HAP)*

To support fast handover for clients roaming from one MAP to another, we propose a method of key pre-distribution among neighboring MAPs. After a home MAP M_1 successfully authenticates a client C through the login authentication protocol, it generates a message containing its ID, the ID of client C, key K_{MAC} and the pairwise master key PMK_0 it shares with the client. The MAP then encrypts the message using the public key P_x of a neighboring MAP M_x, and sends the encrypted message to M_x. (We assume that each MAP has the public key certificates of its neighboring MAPs.) The neighbor MAP M_x decrypts the message using its private key to extract keys K_{MAC} and PMK_0 to prepare for future authentications of client C. The above public key operations are performed by MAPs, which are not constrained in terms of computing capability or power supply.

Since the client may move in any direction, the home MAP should send keys K_{MAC} and PMK_0 to all of its neighbours in anticipation of client C's mobility. The home MAP can combine several encrypted messages (each containing the MAP ID, client ID, K_{MAC} and PMK_0) into one packet and transmit the packet to all neighbours using a broadcast in order to save bandwidth. After a neighbor MAP M_2 receives keys K_{MAC} and PMK_0 and a request for connection from client C, it executes the following handover authentication protocol (presented in the order of the messages exchanged).

$$\begin{array}{l}(1)\ C \longrightarrow M_2\colon\ \Theta_C, N_C, V_{K_{MAC}}(N_C)\\ (2)\ M_2 \longrightarrow C\colon\ N_R, V_{K_{MAC}}(N_C, N_R)\\ (3)\ C \longrightarrow M_2\colon\ N_R, V_{K_{MAC}}(N_R)\end{array}$$

(1) Client C submits its transfer ticket Θ_C to the foreign MAP M_2, along with a nonce N_C, and a message authentication code $V_{K_{MAC}}(N_C)$ to the foreign MAP M_2. The message authentication code is the result of applying the MAC algorithm and secret key K_{MAC} to nonce (N_C).

When M_2 receives this message, it first verifies the correctness of $V_{K_{MAC}}(N_C)$ using the MAC key it received from the home MAP M_1. If the computed MAC value matches $V_{K_{MAC}}(N_C)$, M_2 can confirm that message (1) is valid. Next, M_2 verifies the validity of the transfer ticket. It checks the content of the transfer ticket, especially the ID of the client's ticket agent and the ticket expiry date. It then applies the MAC algorithm and the secret key K_{MAC} received from M_1 to message μ to output a message authentication code $V'_{K_{MAC}}(\mu)$. (Recall from Section 3.3.3 that a transfer ticket consists of two parts: the relevant information stored in a message μ and a message authentication code $V_{K_{MAC}}(\mu)$, which is the result of applying a MAC algorithm and a MAC key to message μ.) If $V'_{K_{MAC}}(\mu) = V_{K_{MAC}}(\mu)$, M_2 can confirm that the transfer ticket is valid (i.e., C was successfully authenticated by its home MAP).

Note that an attacker may capture the transfer ticket and attempt to use it, but will not pass the MAP's authentication, because the attacker cannot produce a valid pair $(N_C, V_{K_{MAC}}(N_C))$ without the knowledge of key K_{MAC}. Furthermore, the pair $(N_C, V_{K_{MAC}}(N_C))$ enables the protocol to resist denial-or-service attacks (see Section 5.6).

(2) M_2 generates a nonce N_R, and computes a message authentication code $V_{K_{MAC}}(N_C, N_R)$, which are sent to client C. When C receives this message, it computes a MAC value $V'_{K_{MAC}}(N_C, N_R)$, using nonces N_C and N_R. If $V'_{K_{MAC}}(N_C, N_R) = V_{K_{MAC}}(N_C, N_R)$, the client has successfully authenticated the foreign MAP. Nonce N_C serves as a challenge C presents to M_2. The inclusion of N_C in the MAC computation is the response of M_2 to the challenge. (We also include nonce N_R in the MAC computation so that the

recipient of the message can detect unauthorized changes to the nonce.)

(3) Client C then executes the MAC algorithm using the MAC key K_{MAC} it computed in step (3) of the log-in authentication (Section 4.1), and the nonce N_R as input. The result is a message authentication code $V_{K_{MAC}}(N_R)$, which C will send to M_2 along with N_R, the challenge from M_2. Upon receiving $N_R, V_{K_{MAC}}(N_R)$, M_2 repeats the same MAC calculation on N_R. If it obtains the same message authentication code as $V_{K_{MAC}}(N_R)$, then this proves C's identity since C is the only client who has the knowledge of the key K_{MAC}.

Following are additional implementation issues and discussions.

(a) If the foreign MAP M_2 receives the transfer ticket Θ_C before the message $r = \{I_C, K_{MAC}, PMK\}$ from the home agent (Section 3.3.3), M_2 will not be able to verify the validity of the transfer ticket because it does not have the MAC key K_{MAC} in order to apply the MAC algorithm to the ticket. In that case, M_2 sends back an error message to C and C who will initiate a log-in authentication instead of handover authentication. In this worst-case scenario, the handover authentication reverts back to the current practice in WMNs, i.e., repeating the login authentication with the foreign MAP. However, with low to moderate mobility speeds, we expect that this worst-case scenario does not happen often, and the handover authentication protocol will be used in most cases.

(b) After M_2 receives message $r = \{I_C, K_{MAC}, PMK\}$ from the home MAP, it also propagates this message to its neighbors to prepare for client C's future move to another MAP, say M_3. M_3 will use message r and the transfer ticket submitted by C to authenticate C as described above.

(c) The handover authentication protocol does not use digital signatures or public key cryptography, but rather a MAC algorithm, to minimize authentication latency during the handover process.

(d) At the end of a successful handover authentication, the foreign MAP and the client will use the PMK to compute a shared key (pairwise transient key PTK) for their subsequent secure communications (see Section 4.3).

(e) The MAC key K_{MAC} has to be updated periodically to maintain its security. When it is updated, the transfer ticket associated with it has to be renewed as well. The MAP R currently serves the client (either a foreign MAP or its home MAP) is responsible for generating a new transfer ticket and a new MAC key. The MAP then encrypts them using the shared key PTK and sends the encrypted message to the client.

(f) During the authentication process, the client and the new MAP also compute a new pairwise master key PMK using the pairwise master key PMK_0 shared by the client and the previous MAP as follows:

$$PMK = f(PMK_0, N_C, N_R) \tag{2}$$

In the above formula, f is a pseudo-random number generation function. N_C and N_R are the nonces generated during the above authentication procedure. Note that, along with the transfer ticket, the client's knowledge of PMK_0 proves to the new MAP that C had been successfully authenticated by another MAP in the network.

Generally speaking, when a client roams from a MAP R_{n-1} to a MAP R_n, the new PMK_n shared by the client and the new MAP is computed using the old PMK_{n-1} shared by the client and MAP R_{n-1}, as follows:

$$PMK_n = f(PMK_{n-1}, N_C, N_R) \tag{3}$$

The client and the new MAP then compute a new PTK using the new PMK, as will be discussed in the next section.

4.3. *Key generation*

We describe briefly the procedure for generating PTKs after a successful authentication between a client C and a MAP R. The PTK generation procedure follows the four-way handshake protocol defined in IEEE 802.11i (IEEE, 2003), as follows.

$$\begin{array}{l}(1)\ R \longrightarrow C:\ M_R, N_R, T_1\\ (2)\ C \longrightarrow R:\ M_C, N_C, T_2, V_{PTK}(M_C, N_C, T_2)\\ (3)\ R \longrightarrow C:\ M_R, N_R, T_3, V_{PTK}(M_R, N_R, T_3)\\ (4)\ C \longrightarrow R:\ M_C, T_4, V_{PTK}(M_C, T_4)\end{array}$$

Notation:

- M_C, M_R: physical addresses of C and R, respectively.
- N_C, N_R: nonces generated by C and R, respectively.
- T_1, T_2, T_3, T_4: message type indicators.

The four-way handshake protocol starts with MAP R generating a nonce N_R and sending it to the client C. Client C receives message (1), generates a nonce N_C, and computes a PTK using the PMK it shares with MAP R as follows.

$$PTK = f(PMK, \min(M_C, M_R) \,\|\, \max(M_C, M_R) \,\|\, \min(N_R, N_C) \,\|\, \max(N_R, N_C)) \tag{4}$$

C then sends a message to R that contains nonce N_C and a message authentication code (MAC) $V_{PTK}(N_C, N_C, MT_2)$. The MAC serves as proof of C's possession of the PMK, because the PTK is the key for generating the MAC and the PTK is computed using the PMK.

Upon receiving message (2), MAP R computes the PTK using Eq. (4), and uses the PTK to verify the MAC sent by C. If the verification is successful, R generates a message authentication code $V_{PTK}(M_R, N_R, MT_3)$ and sends it to C in message (3) so that C can verify R's possession of the PMK. After C successfully verifies the MAC sent by R, it sends a confirmation to R, which is message (4) shown above.

The PTK is updated periodically using the above four-way handshake protocol. The PMK is also updated periodically (but at a much less frequent rate then the PTK) by the login authentication protocol presented in Section 4.1.

5. Security analysis of the proposed authentication protocols

In this section, we identify the security threats (Biryukov et al., 2005; Biryukov and Shamir, 2000; Syverson, 1994)

relevant to our proposed authentication protocols and discuss counter-measures against them.

5.1. Overview

The proposed protocols are protected against various security threats thanks to the following security features:

- Digital signatures of ticket agents in client and MAP tickets: to prevent forgery of and unauthorized modifications to these tickets.
- Public key cryptography: to protect messages (3) and (4) of the login authentication protocol (Section 4.1).
- Symmetric key cryptography: to allow a MAP to securely forward a client's authentication information to another MAP (i.e., message $r = E_{M_1,M_2}(I_C, K_{MAC}, PMK)$ in the handover authentication protocol, Section 4.2, is encrypted with key E_{M_1,M_2} shared by MAPs M_1 and M_2).
- Nonces (used-only-once partial keys): to combat replay attacks and denial-of-service (DoS) attacks, as will be discussed shortly.
- MAC algorithm and MAC keys: to enable a receiver to verify that a message or an information unit (e.g., a nonce) in a message has not been altered in an unauthorized manner. They also provide assurances that a message has been originated by an entity in possession of the MAC key.

The following rule applies to both login and handover authentication protocols:

(R1) A new message with nonces intended for a specific recipient r must use newly generated nonces and not those previously sent to r. If a message with nonces was lost or damaged and the message is retransmitted, the retransmitted message must use newly generated nonces.
(R2) Each message is associated with a timer. If the timer expires before the sender receives a response from the intended recipient of the message, the sender assumes that the message has been lost or damaged.
(R3) If the authentication procedure fails after a predetermined number of tries, the MAP will give up and send the diagnostic information to the network administrator, which will initiate an investigation to determine the cause of the failure.

In addition, a client and a MAP involved in a login authentication session are required to follow the following rule:

(R4) If any of the messages (3) to (6) is lost, the login authentication protocol will restart from step (3).

Similarly, the following rules are required by the handover authentication protocol:

(R5) When a receiver receives a message with a nonce and a corresponding MAC value, it performs the MAC computation. If the resulting MAC value does not match the MAC value in the message, the receiver assumes that this is a message from an attacker.
(R6) If any message of the handover authentication protocol is lost, the protocol will restart from step (1).

Note that message losses and retransmissions discussed in this paper are meant to be associated with the transport layer. (Loss detections and retransmissions may be done at the data link layer [e.g., by the RTS/CTS/DATA/ACK exchange of the IEEE 802.11 medium access control], but are transparent to the authentication protocols and do not follow the above rules.)

In the following sub-sections, we describe the counter-measures implemented in the proposed authentication protocols against the attacks listed in Horn et al. (2002) that are relevant to our protocols.

5.2. Identity privacy attack

Most people would like to remain anonymous while roaming in different parts of network for privacy reasons. To protect clients' privacy, client IDs in tickets are numbers or strings that are not related to the clients' real identities, much like bank account numbers or social security numbers. Only the ticket agents know the mapping between client real identities and client IDs recorded in the tickets they issue.

5.3. Forgery attack

A ticket agent's digital signature ensures that the client tickets it issues are protected against modifications and that counterfeit tickets are infeasible to fabricate.

The integrity of a transfer ticket $\Theta_C = \{\mu, V_{K_{MAC}}(\mu)\}$ is ensured by the accompanying MAC value $V_{K_{MAC}}(\mu)$. Any unauthorized changes to the content of a transfer ticket will result in an incorrect MAC value because the attacker does not know the MAC key shared between the client and its home MAP. Similarly, a counterfeit transfer ticket will not be paired with a correct MAC value due to the counterfeiter's lack of knowledge of the MAC key.

5.4. Time-memory trade-off attack

The simplest form of attack against hash-based MAC algorithms is to use brute force to uncover the secret key. An attacker would use a given input and the corresponding MAC output value (e.g., N_C and $V_{KMAC}(N_C)$ in message (1) of the handover authentication protocol) to figure out the MAC key using brute force. With pre-computation done offline, the time taken in the online stage is shortened at the expense of more memory required. This is called a time-memory trade-off attack. To combat this type of attack, we use current state-of-the-art MAC algorithms, SHA-2, in the proposed protocols, and periodically update MAC keys.

5.5. Replay attack

An attacker records messages of an ongoing authentication session and replays these messages in the future in an attempt to be successfully authenticated and possibly gain access to the network as a client. An attacker may replay a client's messages to gain access to the network, or a MAP's

messages in order to impersonate the MAP. We prevent this type of attack by using message encryption, nonces and the security rules listed in Section 5.1.

5.5.1. *Replaying client messages*

We consider possible replay attacks on messages generated by the proposed authentication protocols.

In the *login* authentication protocol described in Section 4.1, an attacker A overhears and replays message (3) and (5) sent earlier by a client C.

- After successfully receives message (3) from the attacker, the MAP replies with a message (4). New nonces N'_{R1} and N'_{R2} are generated in the message (rule (R1)). The attacker will not be able to decrypt message (4) because he does not know the private key of client C. The attacker then replays message 5 to the MAP. The MAP can detect that this is a replayed message because a new message 5 is supposed to have new nonce N'_{R2} and not N_{R2} in the replayed message 5.

In another attack scenario for attacking the login authentication protocol, an attacker may also replay message (3) or message (5) of a client C.

- If the MAP did not receive the original message (3) from C, the MAP may accept the replayed message as a valid message (if the timer associated with the sent message (2) has not expired yet) and reply with a message (4). However, the attacker will not be able to decrypt message (4) because he does not know the private key of client C, and thus fails to proceed to step (5) of the login authentication protocol. (Client C will also see message (4) sent by the MAP, assuming that it has not timed out on the lost message, and proceed to step (5) of the protocol. In this case, the attacker actually helps instead of harming.)
- If the MAP did not receive the original message (5) from C, the MAP may accept the replayed message as a valid message and reply with a message (6). (Again, the attacker helps the client "retransmit" the lost message (5), assuming that the MAP has not timed out due to the lost message from C.) Note that although the attacker will also receive message (6) it will not be able to access network services because that requires the knowledge of the pairwise master key (PMK) described in Section 4.1. The attacker does not have that knowledge because it does not possess the necessary private keys to decrypt messages (3) and (4) in order to obtain the nonces needed to compute the PMK.

Similarly, to attack the handover authentication protocol proposed in Section 4.2, an attacker captures and replays message (1) and message (3) sent earlier by a client C.

- After successfully receives message (1) from the attacker, the MAP reply with a message (2). A new nonce N'_R is generated in this message (rule (R1)). The attacker then replay message 3. The attacker will not be able to compute the correct MAC value $V_{K_{MAC}}(N'_R)$ because he does not know N'_R and the MAC key K_{MAC}. The attacker thus fails the authentication by the MAP in step (3) (rule (R5)).

In another attack scenario for the handover authentication protocol, an attacker may also replay a message (1) or message (3) of a client C.

- If the MAP did not receive the original message (1) from C, it may accept the replayed message as a valid message and reply with a message (2). However, the attacker will not be able to compute the correct MAC value $V_{K_{MAC}}(N'_R)$ because it does not know the MAC key K_{MAC} and thus fails the authentication by the MAP in step (3) (Rule (5)).
- If the MAP did not receive the original message (3) from C, it may accept the replayed message as a valid message. The client is then considered successfully authenticated by the MAP, assuming that the MAP receives the replayed message before it times out on the lost message. The attacker, on the other hand, will not be able to be authenticated by the MAP because it does not know the PMK shared between the client and its home MAP and known by the foreign MAP. Thus, the authentication fails and the attacker cannot get access to the networks.

5.5.2. *Replaying MAP messages*

We examine possible attack scenarios aimed at replaying MAP messages. In the login authentication protocol described in Section 4.1, an attacker overhears and replays message (4) and (6) sent earlier by a MAP R.

- The client sends message (3), $E_{P_R}(\{T_C, N_{C1'}, N_{C2'}\})$, to the MAP. New nonces N'_{C1} and N'_{C2} are generated in the message (Rule (1)). The attacker replays message (4) and the client reply with a message (5). The attacker then replays message 6, N_{C2}, Θ_C. The MAP can detect that this is a replay attack because message 6 is supposed to have new nonce N'_{C2} and not N_{C2} in the replayed message (rule (R1)). The attacker will not be able to decrypt message (3) and get the new nonce N'_{C2} because he does not know the private key of MAP R.

In another attack scenario for attacking login authentication protocol, an attacker may replay message (4) or message (6) sent earlier by a MAP R.

- If the client did not receive the original message (4) from R, the client may accept the replayed message as a valid message (if the timer on the sent message (3) has not expired yet), and reply with a message (5). However, the attacker will not be able to generate the MAC value $V_{KMAC}(N_{C2})$ because he does not know the MAC key K_{MAC}, and thus fails the authentication by the client in step (6).

Note that the MAP may also receive the replayed message correctly and proceed to step (6) of the protocol. In this case, the attacker actually helps to "retransmit" the message (4) that the MAP lost in the first place. (If R does not receive the replayed message (5), client C will time out on waiting for message (6) from R and restart the authentication procedure.)

- If client C did not receive the original message (6) from the MAP, C will accept the replayed message and consider the authentication successful (assuming that C receives the replayed messages before it times out on the lost message). However, the attacker will not be able to impersonate the

MAP because it does not know the PMK shared by the client and the MAP, which is required for subsequent communications between the client and the MAP.

Similarly, to attack the handover authentication protocol proposed in Section 4.2, an attacker overhears and replays a message (2) sent earlier by a MAP R.

The client sends message (1), $\Theta_C, N'_C, V_{K_{MAC}}(N'_C)$, to the MAP. The client generates a new nonce N'_C in message 1 (rule (R1)). The attacker replays message (2), $N_R, V_{K_{MAC}}(N_C, N_R)$. The attacker will not be able to compute the correct MAC value $N_R, V_{K_{MAC}}(N'_C, N_R)$, because it does not know the MAC key K_{MAC} and thus fails the authentication by the MAP in step (2) (rule (R5)).

In another attack scenario for attacking handover authentication protocol, an attacker may replay a message (2) sent earlier by a MAP R.

If the client did not receive the original message (2), it may accept the replayed message as a valid message and reply with a message (3) (before it times out on the lost message). The MAP is considered successfully authenticated by the client. The attacker, however, will not be able to communicate with the client because it does not have the knowledge of the PMK, as discussed above.

5.6. *Denial-of-service (DoS) attack*

An attacker may send bogus messages or replay past valid messages repeatedly to force a MAP to spend resources on processing a large amount of these DoS attack messages. To combat DoS attack, the proposed authentication protocols rely on the security features and rules stated in Section 5.1.

5.6.1. *Analysis of the login authentication protocol*

An attacker may repeatedly send copies of message (1) to a MAP. The MAP will interpret the duplicates of this message as the losses of messages (2) it has sent. The MAP will stop the authentication procedure after a pre-determined number of failed attempts according to rule (R(3)) to save resources. Note that this type of attack can happen to any protocol, and not specifically to authentication.

An attacker may sniff valid message (3) and message (5) from a successful login authentication and replay the message repeatedly to the involved MAP in order to overwhelm it. The MAP can detect that this is a replayed attack because a new message 5 is supposed to have new nonce. If the MAP receives the replayed message several times, it can infer that it is under a DoS attack and take appropriate actions to thwart the attack (Aura et al., 2000; Wang and Reiter, 2003; Lemon, 2002).

Note that an attacker may flood a MAP with bogus copies of message (3) that it creates by itself, but those bogus messages will be detected by the MAP because the attacker could not possess a valid client ticket T_C. After processing a number of such bogus messages, the MAP can infer that it is under a DoS attack and take appropriate actions. (If an attacker possesses a valid client ticket, this can be categorized as an insider attack, which is much harder to detect. This requires human interventions, e.g., checking if the mobile device was stolen; verifying the client's background.).

5.6.2. *Analysis of the handover authentication protocol*

All messages of the handover authentication protocol are protected against forgery and unauthorized modifications by the MAC algorithm. An attacker cannot generate a valid message in the handover authentication protocol without the knowledge of the MAC key shared only by the client, its home MAP and the foreign MAP (Rule (R5)).

On the other hand, an attacker may repeatedly replay a message (1) (or message (3)) originated earlier by a client C. The MAP can detect that these are replayed messages because the attacker will not be able to compute the correct MAC value $V_{K_{MAC}}(N'_R)$ and thus fails the authentication by the MAP in step (3). If the MAP receives the replayed message several times, it can conclude that it is under a DoS attack and take necessary counter-attack measures (Xu and Wang, 2007; Keromytis et al., 2004; Waters et al., 2004).

5.7. *Compromised MAPs*

An attacker may compromise a MAP by (1) dropping valid authentication messages to prevent clients from joining the network, or (2) granting access to unauthorized/non-paying users. Following are effective counter-measures against these attacks.

(1) Dropping valid messages deviates from the normal procedure of the authentication protocol, which requires the attacker to modify the authentication code. Software-based attestation techniques such as SWATT (Seshadri et al., 2004) and Pioneer (Seshadri and Luk, 2005) can be used to externally verify the contents of the memory of an embedded device (SWATT) or a CPU (Pioneer) in order to detect changes to the original code. An external verifier can detect with high probability if a single byte of the memory deviates from the expected value (Seshadri et al., 2004). These techniques allow a network operator to periodically verifies the routers in its network and detect compromised nodes. Note that this attack can happen to any protocols (e.g., routing) and not just authentication. From a client's point of view, the attack consequence is similar to that of a router failure: the client times out on the authentication request, and will look for another MAP nearby to join. This type of router placement redundancy should be implemented regardless of security issues: if a MAP fails or malfunctions, nearby MAPs should be able to support its clients.

(2) To grant access to users that do not own valid tickets, the attacker would need to modify the authentication code. Thus one countermeasure is to use attestation techniques such as SWATT and Pioneer to detect changes in the authentication code, as discussed above. An alternative we propose is to use a dual authentication process. The authentications described in Section 4, if successful, give the client only short-term access to network services. The client will subsequently be authenticated by an authentication server (via multi-hop communications), while enjoying network services using the short-term access permission. After the server successfully authenticates the client, it will issue to the client a service ticket (Kohl and Neuman, 1993) that serves as a pass for the client to

access network services on a long-term basis. An illegitimate or non-paying user will not be issued such a service ticket, and will not be able to continue to use network services after the short-term access privilege expires. The dual authentication process allows both fast authentication during the handover step, and the stronger security provided be an authentication server.

6. Performance evaluation

We compare the performance of our proposed authentication protocols with existing protocols using both numerical analysis and simulations. The protocols to be compared include EAP-TLS and the algorithm proposed by Kassab (2007). EAP-TLS is a popular authentication protocol for IEEE 802.11-based wireless networks and represents the multi-hop handover authentication approach. Kassab's (Kassab, 2007) and Li's (Li, 2010) algorithms are representative of the ticket-based approach and the closest to ours. Kassab's and Li's algorithms work in a similar manner. The major difference between them is that the authentication server (AS) in Kassab's distributes PMKs to the MAPs adjacent to the home MAP, while the AS in Li's distributes tickets. The size of a ticket is bigger than that of a PMK. Thus the traffic overhead incurred by Li's is higher than that by Kassab's. Therefore we chose to compare our handover authentication protocol (HAP) with the more efficient algorithm, Kassab's.

6.1. Numerical analysis

The numerical analysis demonstrates the theoretical gain of our proposed protocols over EAP-TLS and Kassab's scheme. The performance of the protocols is measured in terms of

- *communication costs*, which indicate the number of messages exchanged between a MAP and a client to complete an authentication session.
- *computation costs*, which are the latencies (in milliseconds) incurred by the following security operations: encryption using public key (E_{pub}); decryption using public key (D_{pub}); encryption using shared key (E_K); decryption using shared key (D_K); generation of a digital signature (G_{sig}); verification of a digital signature (V_{sig}); computation/verification of a message authentication code (MAC); and hashing.

Table 3 lists the above operations, the current state-of-the-art algorithms implementing the operations, and the computation time each of these algorithms incurs (Long, 2006) (the first, second and third columns, respectively). The fourth, fifth and sixth columns of Table 3 list the numbers of security operations the proposed login and handover authentication protocols, Kassab's scheme and EAP-TLS perform, respectively. By multiplying the computation cost of each operation (from the third column) and the number of times it is executed, and summing up the costs of all operations executed by a protocol, we obtain its total computation cost as shown in the third last row of Table 3. The computation cost of the login authentication protocol (97.935 ms) is slightly less than that of EAP-TLS (97.962 ms). But more importantly, the computation cost of the handover authentication protocol (0.105 ms) is 2.45% of the Kassab's scheme (4.3 ms) and is three orders of magnitude lower than that of the login authentication and EAP-TLS.

The second last row of Table 3 lists the number of messages exchanged in each protocol. The authentication latencies shown in the last row are the sums of computation costs and communication delays, where d is the average delay of a one-hop transmission incurred by a message, and h is the number of hops between the client and the home authentication server. (Parameter h is applicable to only EAP-TLS as our handover protocol and Kassab's handover scheme does not require a client to communicate with the home MAP during the hand-off process.) The average delay of a one-hop transmission d includes the backoff time, RTS/CTS/DATA/ACK exchange and DIFS and SIFS values, transmission time, propagation time and processing time as shown in Fig. 4. The results show that the larger the number of hops between a client's home MAP and a foreign MAP, the lower the authentication latency our protocols incur compared with EAP-TLS.

In particular, the gain of the login authentication protocol over EAP-TLS is due to

- a reduction in the number of messages exchanged, six vs. nine;
- one-hop communication between the client and the MAP vs. multi-hop communication between client and the authentication server (captured by parameter h).

The gain of the handover authentication protocol over EAP-TLS is also due to the above two reasons, plus the elimination of public key operations during the handover authentication. The gain of the HAP over Kassab's protocol results from less cryptographic operations, and one less message, three vs. four.

6.2. Simulation results

We use QualNet (version 4.5), a commercial software that provides scalable simulations of wireless networks (QualNet Simulator), for our experiments.

6.3. Performance metrics

One performance metric is *authentication delay* (latency), which is measured as the time between a client's transmission of an authentication request to a nearby MAP and the receipt of an acceptance confirmation. After a client sends an authentication request, it sets a timer. If it does not receive a confirmation by the time the timer expires, it will re-send the request. The authentication delay is measured starting with the first request. In all experiments, we calculate the *average authentication delay* (AAD), averaged over all mobile clients participating in the experiment. In several cases, we also keep track of the *maximum authentication delay* (MAD), the maximum value among all mobile clients.

In the proposed HAP, after a successful login authentication, the home MAP will send the client's transfer ticket, the PMK and a MAC key it shares with the client to the neighboring MAPs to prepare for a handover in a near future. (This is a one-hop communication, from the home MAP to the

Table 3 – Computation and communication costs.

Op.	Alg.	Time (ms)	Login see 4.1	Handover see 4.2	EAP-TLS	Kassab's
E_{pub}	RSA (Rivest et al., 1978)	1.42	1	0	1	0
D_{pub}	RSA	33.3	1	0	1	0
G_{sig}	ECDSA (ECDSA, 2009)	11.6	1	0	1	0
V_{sig}	ECDSA	17.2	3	0	3	0
E_K	AES	2.1 (Sterbenz and Lipp, 2000)	0	0	0	1
D_K	AES	2.2 (Sterbenz and Lipp, 2000)	0	0	0	1
MAC	HMAC (Krawczyk et al., 1997)	0.015	1	7	0	2
Hash	SHA-2 (Manuel, 2011)	0.009	0	0	3	0
Total computation cost (ms)			97.935	0.105	97.962	4.3
Number of messages			6	3	9	4
Authentication latency (ms)			97.935+6d	0.105+3d	97.962+9dh	4.3+4d

neighboring MAPs in one broadcast message.) In Kassab's protocol, after a successful login authentication, the authentication server (AS) sends to every neighbor N of the home MAP a PMK to be shared by N and the client when the client roams and needs to be authenticated by N. (These are multi-hop communications, from the AS to each neighboring MAP.) These pre-distributions of keys/tickets incur some delay before the next handover. We call this delay *key pre-distribution delay*, which should be minimized to avoid service interruption when clients move from one MAP to another. We compare the proposed HAP with Kassab's protocol in terms of key pre-distribution delay.

6.4. *Simulation parameters*

The common simulation parameters for all experiments are listed in Table 5. The transmission range of the wireless routers (MAPs) is 315 m, according to the specifications of wireless routers manufactured by TROPOS. The transmission range of mesh clients is 304 m, according to the specifications of wireless adapter manufactured by Cisco. The transmission rate at the physical layer is 2 Mbits/s. Mobility speeds of mobile clients vary from 0 to 30 m/s and the mobility pattern follows the random way point model (Broch et al., 1998). Each data point in the graphs is the average of 10 runs using different random seeds. The graphs are plotted with a confidence interval of 95%.

We conducted eight sets of experiments:

1. We measured the average authentication latency of the login authentication protocol (LAP) as a function of clients' mobility speed. The 400 m × 400 m network has one MAP placed in the center of the square. Three scenarios: 20, 40 and 60 clients. In each experiment, all clients have the same mobility speed. The speed is varied from 0 m/s to 30 m/s.
2. We compared the LAP with EAP-TLS and measured both the AAD and MAD. We used the same network as in experiment (a). All clients moved at the same speed of 20 m/s. The number of clients varied from 10 to 60.
3. We measured the AAD of the handover authentication protocol (HAP) as a function of clients' mobility speed. We simulated a network of size 600 m × 600 m with four MAPs arranged as in Fig. 5, and three scenarios: 20, 40 and 60 clients in the network, respectively. In each experiment, all clients have the same mobility speed. The speed is varied from 0 m/s to 30 m/s.
4. We measured the AAD and MAD of the handover authentication protocol (HAP) as functions of number of clients. We used the same network as in experiment (3). All clients moved at the same speed of 20 m/s. The number of clients varied from 10 to 60.
5. We compared the HAP with EAP-TLS and Kassab's algorithm in terms of the average authentication delay (ADD) during the handover process. We used the network configuration shown in Fig. 6. The home MAP H has four neighbor MAPs. The authentication server was located six hops away from each MAP in order to illustrate the high overhead of the multi-hop handover authentication approach used by EAP-TLS. We varied the number of clients from 10 to 60. All clients moved at the same speed of 20 m/s.

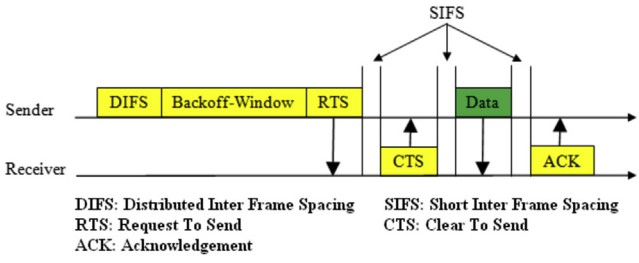


Fig. 4 – Delay incurred by a one-hop transmission.

Table 4 – Common simulation parameters.

Parameter	Value
Movement model	Random way point
Speed	0–30 m/s
Propagation fading model	None
Transmission range of MAPs	315 m
Transmission range of mesh clients	304 m
Transmission rate at physical layer	2 Mbits/s
Physical layer protocol	PHY802.11b
Number of runs per data point	10
Confidence interval	95%

Table 5 – Simulation parameters for different experiment.

Experiment	Network	Clients, mobility speed
1. Fig. 7(a), LAP	400 m × 400 m, one MAP	20–60 nodes, 0–30 m/s
2. Fig. 7(b), LAP vs. EAP-TLS		10–60 nodes, 20 m/s
3. Fig. 7(c), HAP	600 m × 600 m, four MAPs	20–60 nodes, 5–30 m/s
4. Fig. 7(d), HAP		10–60 nodes, 10–20 m/s
5 and 6. Fig. 7(e)–(h), HAP vs. Kassab, and EAP-TLS	600 m × 600 m, five MAPs, each MAP is six hops away from the AS	10–60 nodes, 20 m/s
7 and 8. Fig. 7(i) and (j), HAP vs. Kassab	600 m × 600 m, five MAPs, each MAP is six hops away from AS	10–60 nodes, 20 m/s

6. This experiment is the same as experiment (5) above, except that we recorded the maximum authentication delay (MAD) during the handover.
7. We compare the HAP with Kassab's algorithm in terms of the average key pre-distribution delay. The network and simulation parameters are the same as those in experiment (5) above.
8. This experiment is the same as experiment (7) above, except that we recorded the maximum key pre-distribution delay.

The simulation parameters specific to each experiment are summarized in Table 4. In all the experiments, the mobile clients were randomly distributed in the networks. To test the scalability of the protocols, we let all clients present in the network send authentication requests to their respective nearby MAPs *simultaneously*.

6.5. *Result analysis*

The results of the above eight sets of experiments are illustrated by the graphs in Fig. 7.

1. The graph in Fig. 7(a) shows the AAD of the login authentication protocol (LAP) as a function of clients' mobility speed. There is one MAP placed at the center of the network, serving 10–60 mobile clients. Each client is one hop away from the MAP. We observe that the AAD is not impacted much by the mobility speed, which is a positive attribute of the LAP. On the other hand, as the number of clients increases from 20 to 60, the ADD also increases as expected, by approximately 4%–6%. More clients imply more authentication requests to be processed by the MAP, and more channel contention around the MAP, resulting in longer delay.
2. Fig. 7(b) shows the performance of the login protocol vs. EAP-TLS under the same network setting as above. When there are only 10 clients in the network, both protocols perform similarly. Given more than 10 clients, the workload and channel contention at the MAP increases. In these cases, the LAP offers lower AAD than EAP-TLS, because the LAP requires less messages exchanged than EAP-TLS (6 vs. 9, as shown in the second last row of Table 3). In the case of 60 clients, the AAD of the LAP is 16% lower than that of EAP-TLS. As the number of nodes increases, the performance gap between the LAP and EAP-TLS enlarges, consistent with the authentication latencies recorded in the last row of Table 3 ($97.935+6d$ vs. $97.962+9dh$, where $h = 1$). The graph also shows the MAD of both protocols. The MAD of the LAP is about 32% higher than its AAD, which we deem acceptable, and about 20% lower than the MAD of EAP-TLS.

Given 60 mobile clients connecting through the same MAP, the MAD of LAP and EAP-TLS are 321.7 ms and 387.6 ms, respectively, or LAP improves the login authentication delay by 65.9 ms. The amounts of cryptographic computation performed by LAP and EAP-TLS are very similar (97.935 ms vs. 97.062 ms as shown in the last row of Table 3). This shows that the gain of LAP over EAP-TLS is mainly due to one-hop communication between the client and the home MAP in LAP versus multi-hop communication between the client and the authentication server (AS) in EAP-TLS and to the reduction of the number of messages exchanged from nine to six.

3. The graph in Fig. 7(c) shows the AAD of the handover authentication protocol (HAP) as a function of clients' mobility speed. Four MAPs are uniformly distributed over

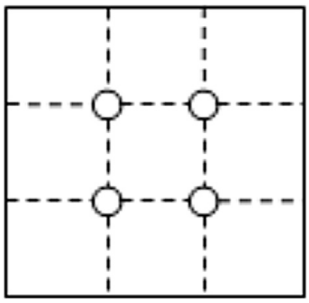


Fig. 5 – Network with four MAPs.

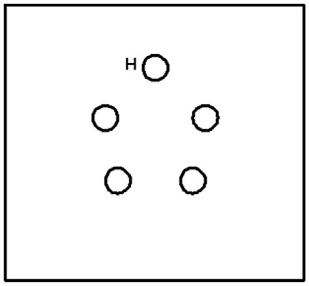
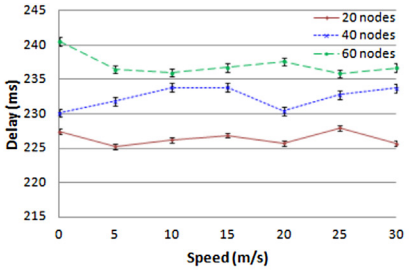
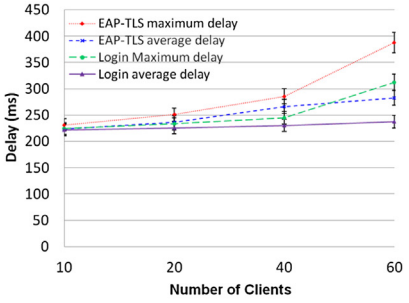


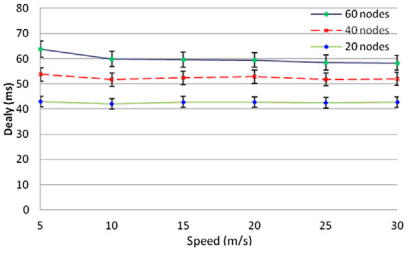
Fig. 6 – Network with five MAPs.



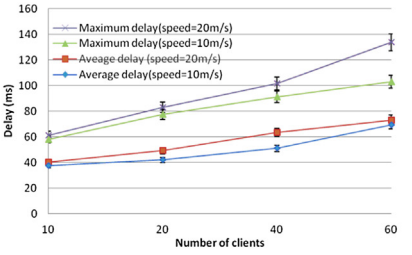
(a) Login protocol - Function of mobility speed



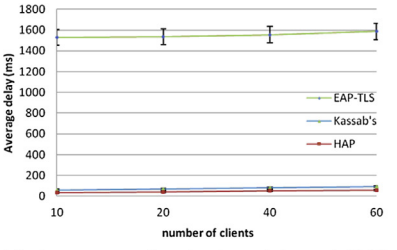
(b) Login protocol vs. EAP-TLS - Function of number of clients



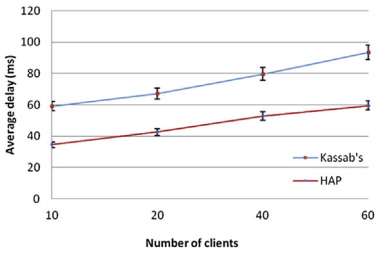
(c) Handover protocol - Function of mobility speed



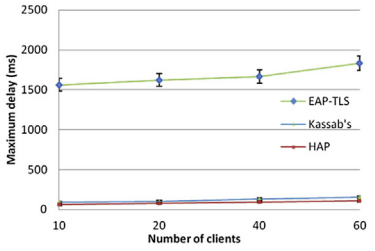
(d) Handover protocol - Function of number of clients



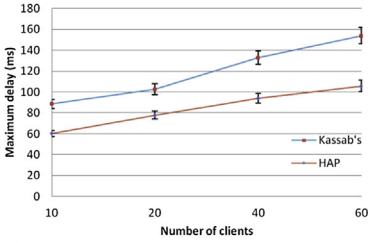
(e) Average authentication delay of EAP-TLS, Kassab's protocol and HAP - Function of number of clients



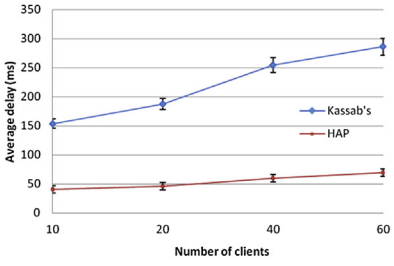
(f) Average authentication delay of HAP vs. Kassab's protocol - Function of number of clients



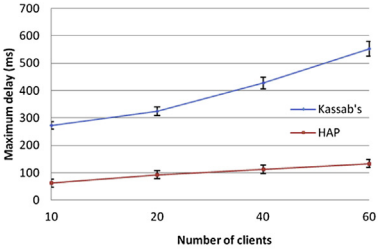
(g) Maximum authentication delay of EAP-TLS, Kassab's protocol and HAP - Function of number of clients



(h) Maximum authentication delay of HAP vs. Kassab's protocol - Function of number of clients



(i) Average key pre-distribution latency of HAP vs. Kassab's protocol- Function of number of clients



(j) Maximum key pre-distribution latency of HAP vs. Kassab's protocol - Function of number of clients

Fig. 7 – Simulation result.

the network, serving 10–60 mobile nodes. Again, the mobility speed does not have a big impact on the AAD of the HAP, as in the case of the LAP. Also, the more clients send requests, the higher the AAD, as expected. Note very low AADs of the HAP, ranging from 42.96 ms to 63.8 ms, compared with the AADs of the one hop LAP and one hop EAP-TLS which are above 220 ms.

4. The above observations also apply to Fig. 7(d), which shows the MADs and AADs of the handover authentication protocol as functions of number of clients. In the experiment with 60 nodes moving at a speed of 10 m/s, the MAD of the HAP is 103.2 ms, about 150% of the corresponding AAD, but still very low compared with the authentication delay of EAP-TLS.
5. The graph in Fig. 7(e) shows the AAD of EAP-TLS, Kassab's protocol and the HAP as functions of the number of clients given the network topology in Fig. 6. The AS is six hops away from the home MAP. As the number of clients increases from 10 to 60, the AAD of all three schemes increases as expected due to higher loads on the MAPs and more traffic in the network. Both the Kassab's protocol and the HAP outperform EAP-TLS by a large margin in terms of AAD, thanks to one-hop communication (between the client and the foreign MAP) during the handover authentication versus multi-hop communication (between the client and the AS) done by EAP-TLS. Moreover, the AAD of the HAP is much lower than that of EAP-TLS due to a reduction in the number of messages exchanged, three vs. nine (see the second last row of Table 3).

We separated the curves of the HAP and Kassab's protocol from Fig. 7(e) and magnified them in Fig. 7(f). The new graph shows that the HAP noticeably outperforms Kassab's protocol. For example, when the number of clients is 60, the AADs of HAP and Kassab's scheme are 59.5 ms and 93.3 ms, respectively. HAP improves the authentication delay by 33.8 ms or 57% compared to Kassab's scheme, out of which a reduction of 4.3 ms is due to less cryptographic computation. Kassab's algorithm requires three more decryption operations and one more encryption than HAP (see the third last row of Table 3). The remaining 29.4 ms (74.26%) authentication delay improvement results from the HAP incurring less message exchanges than Kassab's, three vs. four (see the Appendix).

6. The above observations and explanations also apply to the graphs in Fig. 7(g) and (h), which show the MAD of EAP-TLS, Kassab's protocol and the HAP as functions of the number of clients. In all cases, the HAP incurs lower MAD than both EAP-TLS and Kassab's protocol.
7. Fig. 7(i) shows the average key pre-distribution delay (KPDD) of the HAP and Kassab's scheme. As the number of clients increases from 10 to 60, the average KPDD ranges from 273.3 ms to 552.8 ms for Kassab's protocol, and from 61.7 ms to 133.8 ms for the HAP. That is, the average KPDD of HAP is from 55% to 50.3% lower than that of Kassab's scheme. A lower KPDD implies less service interruption, because neighboring MAPs are prepared earlier to connect with a roaming client.

Given 60 mobile clients trying to join the network via the same MAP, the average KPDDs of HAP and Kassab's scheme are 133.8 ms and 552.8 ms, respectively. The HAP improves the average KPDD by 419 ms comparing to Kassab's scheme. The computation cost of HAP key pre-distribution is n encryptions, where n denotes the number of MAPs adjacent to the home MAP. The computation cost of Kassab's key pre-distribution is $2n + 2$ encryptions and 2 decryptions (see Table 3). Given $n = 4$ in this experiment and assuming that the cryptographic operations are performed one after another,[2] the computation cost of the HAP is 17 ms less than that of Kassab's. The remaining 200.1 ms (96.17%) out of 419 ms PKDD improvement by the HAP result from the use of transfer tickets, which eliminate multi-hop communications between the authentication server and the neighboring MAPs, and from the reduction of one message exchanged ($2n$ messages in the HAP vs. $2n + 1$ messages in Kassab's).

8. The maximum key pre-distribution delays (KPDD) of the HAP and Kassab's scheme are shown in Fig. 7(j). The above observations and explanations apply to this experiment as well. In short, the HAP offers lower maximum KPDD compared to than Kassab's protocol, from 55% to 50.3% lower. Almost all the gain of the HAP over Kassab's (95%) is the result of the use of transfer tickets to avoid multi-hop communications between the authentication server and the neighboring MAPs.

Both the performance analysis and simulation results confirm the advantage of the proposed LAP over the EAP-TLS protocol of IEEE 802.11s and the HAP over Kassab's protocol and EAP-TLS. This contributes toward a faster hand-off process for mobile clients using real-time services in WMNs.

7. Conclusion

The objective of our work is to extend the capabilities of IEEE 802.11s standards to support fast hand-off for real-time applications such as VoIP, tele-conferencing, and stock quote distributions. We propose new authentication protocols to support fast login and hand-off in IEEE 802.11s networks. A client and a MAP mutually authenticate each other using one-hop communications. Fast authentication for roaming from one MAP to another is supported by using transfer tickets. The authentication server is not required to participate during the handover authentication process (but only after the client has joined the new MAP if dual authentications are implemented). Our numerical analysis and simulation results confirm that the proposed LAP and HAP outperform the EAP-TLS protocol of IEEE 802.11s and a representative of the ticket-based authentication approach, Kassab's protocol. They are also resilient to various kinds of attacks. In our future work, we will extend the proposed protocols to support multiple network operators and multiple ticket agents, and evaluate the performance of the dual authentication approach.

[2] In practice, the neighboring MAPs may perform the cryptographic operations in parallel after receiving the key(s).

Appendix

Kassab's handover authentication protocol

Before a client *C* moves from a serving MAP M_1 to a target MAP M_2, *C* generates a ticket for M_2 and forwards it to M_1. M_1 will forward the ticket to M_2. Following is the structure of the ticket:

$$E_{IAPPkey}(I_C; E_{PMK}(I_C; K))$$

The ticket contains *C*'s ID and a key *K* which *C* will share *K* with M_2 after a successful authentication. Both *C*'s ID and *K* are encrypted with a pairwise master key *PMK* shared by *C* and M_2 and pre-distributed by the authentication server to *C* and M_2. The encrypted message is then concatenated with *C*'s ID and the content of the ticket is encrypted again with an IAPP (Inter-Access Point Protocol) key (Garcia et al., 2006) shared by M_1 and M_2. After M_2 receives the ticket from M_1, it decrypts the message using the shared IAPP key and the PMK to obtain key *K* to prepare for future authentication of client *C*.

The authentication protocol is executed as follows (presented in the order of the messages exchanged):

$$\begin{array}{l}(1)\ C \longrightarrow M_2\text{: } I_C, I_M\\ (2)\ M_2 \longrightarrow C\text{: } ACK\\ (3)\ C \longrightarrow M_2\text{: } N_C, I_C, E_{KEK}(K_{Share}), V_{K_{MAC}}(N_C, I_C, E_{KEK}(K_{Share}))\\ (4)\ M_2 \longrightarrow C\text{: } I_M, V_{K_{MAC}}(I_M)\end{array}$$

(1) Client *C* submits its ID and M_1's ID to M_2. When M_2 receives this message, M_2 generates two keys using the shared key *K*, a key encryption key *KEK* and a MAC key K_{MAC}.
(2) M_2 replies with an acknowledgment (ACK). After *C* receives the acknowledgment, *C* generates the same two keys *KEK* and K_{MAC} using the shared key *K*.
(3) Client *C* generates a nonce N_C and encrypts a new key K_{Share} with key *KEK*. *C* sends N_C, the encrypted K_{Share} key, along with *C*'s ID and a message authentication code $V_{K_{MAC}}(N_C, I_C, E_{KEK}(K_{Share}))$ to M_2. When M_2 receives this message, it computes a MAC value

$$V'_{K_{MAC}}(N_C, I_C, E_{KEK}(K_{Share})).$$

If $V'_{K_{MAC}}(N_C, I_C, E_{KEK}(K_{Share})) = V_{K_{MAC}}(N_C, I_C, E_{KEK}(K_{Share}))$, M_2 has successfully authenticated the client *C*. M_2 decrypts $E_{KEK}(K_{Share})$ and obtain the same key K_{Share} as client *C*.
(4) M_2 sends its ID I_M along with the MAC value $V_{K_{MAC}}(I_M)$ to *C*. Upon receiving the message, *C* repeats the same MAC calculation on I_M. If it obtains the same message authentication code as $V_{K_{MAC}}(I_M)$, then this proves M_2's identity since M_2 is the only party in the network that has the knowledge of key K_{MAC}.

REFERENCES

Amir Y, Danilov C. Fast handoff for seamless wireless mesh networks. ACM MobiSys; 2006.

Aura T, Nikander P, Leiwo J. Dos-resistant authentication with client puzzles. In: The 8th international workshop security protocols, Cambridge 2000.

Biryukov A, Shamir A. Cryptanalytic time/memory/data tradeoffs for stream ciphers. In: Advances in cryptology: proceedings of ASIACRYPT. Lecture notes in computer science 2000;vol. 1976.

Biryukov A, Mukhopadhyay S, Sarkar P. Improved time-memory trade-offs with multiple data. In: 12th Annual workshop on selected areas in cryptography. Lecture notes in computer science. Springer; 2005.

Broch J, Maltz DA, Johnson DB, Hu Y, Jetcheva J. A performance comparison of multi-hop wireless ad hoc network routing protocols. In: ACM international conference on mobile computing and networking (MOBICOM) 1998.

Buddhikot M. Design and implementation of a WLAN/CDMA 2000 inter-networking architecture. IEEE Communication Magazine; 2003.

Cisco Aironet 802.11 Wireless Adapter. http://www.cisco.com/.

Draft amendment: ESS mesh networking 2009. IEEE 802.11s IEEE802.11s/D3.0.

Du W, Deng J, Han Y, Varshney P, Kate J, khalili A. A pairwise key pre-distribution scheme for wireless sensor networks. In: ACM conference on computer and communications security (CCS 03) 2003.

ECDSA, FIPS 186-3, Digital Signature Standard (DSS) 2009.

Forsberg D, Ohba Y, Patil B, Tschofenig H. Protocol for carrying authentication and network access (PANA) 2008. RFC 5191.

Fu A. A fast handover authentication mechanism based on ticket for 802.16 m. IEEE Communication Letter 2010;14(12).

Garcia E, Faixo L, Vidal R, Paradells J. Inter-access point communication for distributed resource management in 802.11 networks. In: WMASH 2006 2006.

3GPP Technical Specification 22.121 v5.3.1: "The virtual home environment (release 5)" 2002.

Hiertz GR. IEEE 802.11s: the WLAN mesh standard. IEEE Wireless Communications 2010;17(1).

Horn G, Martin M, Mitchell C. Authentication protocols for mobile network environment value-added services. IEEE Transaction on Vehicular Technology 2002;51(2).

Huang P. A fast handoff mechanism for IEEE 802.11 and IAPP networks. IEEE ITC; 2006.

IEEE. Part11: wireless medium access control (MAC) and physical layer specifications: medium access control (MAC) security enhancement 2003. IEEE Standard 802.11i/D10.0.

Jablon DP. Password authentication using multiple servers. In Topics in cryptology 2001.

Jiang Y, Lin C, Shen X, Shi M. Mutual authentication and key exchange protocols for roaming services in wireless mobile networks. In: IEEE transaction on wireless communications 2006.

Kassab M. Fast pre-authentication based on proactive key distribution for 802.11 infrastructure networks. In: ACM WMuNeP 2005.

Kassab M. Securing fast handover in WLANs: a ticket based proactive authentication scheme. In: Globecom2007 2007.

Keromytis AD, Misra V, Rubenstein D. SOS: an architecture for mitigating DoS attacks. IEEE Journal of Selected Areas in Communications 2004;22(1).

Kohl J, Neuman C. The Kerberos network authentication service (V5) 1993. RFC 1510.

Krawczyk H, Bellare M, Canetti R. HMAC: keyed-hashing for message authentication 1997. RFC 2104.

Lemon J. Resisting SYN flood DoS attacks with a SYN cache. In: BSDCON 2002 2002.

Li G. A ticket-based authentication scheme for fast handover in wireless local area networks. In: Wireless communications networking and mobile computing (WiCOM) conference 2010.

Li C, Nguyen UT. Fast authentication for mobile hosts in wireless mesh networks. Technical Report. Department of Computer Science & Engineering, York University; 2010.
Long M. Energy-efficient and intrusion resilient authentication for ubiquitous access to factory floor information. In: IEEE transaction on industrial informatics 2006.
Manuel S. Classification and generation of disturbance vectors for collision attacks against SHA-1. Designs, Codes and Cryptography 2011;59(3).
Marenic T. Designing reference architecture for providing virtual home environment. In: ConTEL 2003 2003.
Mishra A, Shin M, Petroni NL, Clancy TC, Arbaugh W. Pro-active key distribution using neighbour graphs. IEEE Wireless Communications 2004;11(1).
Park C, Hur J, Kim C, Shin Y, Yoon H. Pre-authentication for fast handoff in wireless mesh networks with mobile APs. In: WISA'06, information security applications 2007.
Pizada AA, McDonald C. Kerberos assisted authentication in mobile ad-hoc networks. In: Conference on Australian computer science 2004.
Qazi S. Securing wireless mesh networks with ticket-based authentication. In: Signal processing and communication system 2008.
QualNet Simulator, http://www.scalablenetworks.com/.
RFC 2104 – HMAC: Keyed-Hashing for Message Authentication.
Rivest R, Shamir A, Adleman L. A method for obtaining digital signatures and public key cryptosystems. In: Communication of the ACM 1978.
Seshadri A, Luk M. Verifying code integrity and enforcing untampered code execution on legacy system. In: 20th ACM symposium on operating systems principles 2005.
Seshadri A, Perrig A, van Doorn L, Khosla P. SWATT: software-based attestation for embedded devices. In: IEEE symposium on security and privacy 2004.
Shi M. A ticket ID system for service agent based authentication in WLAN/cellular integrated networks. WCNC; 2007.
Srivatsa AM, Xie J. A performance study of mobile handoff delay in IEEE 802.11-based wireless mesh networks. In: 2008 IEEE international conference on communications 2008.
Sterbenz A, Lipp P. Performance of the AES candidate algorithms in Java. In: The third advanced encryption standard candidate conference, New York, USA 2000. p. 161–5.
Syverson P. A taxonomy of replay attacks. In: Computer security foundations Symposium. IEEE Computer Society Press; 1994.
TROPOS networks. http://www.tropos.com.
Velayos H, Karlsson G. Techniques to reduce IEEE 802.11b MAC layer handover time. KTH Technical Report. ISSN1651-7717 2003. ISRN KTH/IMIT/LCN/R-03/02SE, Stockholm, Sweden.
Wang X, Lim AO. IEEE 802.11s wireless mesh networks: framework and challenges. Ad Hoc Networks Journal (Elsevier) 2008;6(6).
Wang X, Reiter MK. Defending against denial-of-service attacks with puzzle auctions (extended abstract). In: IEEE symposium on security and privacy 2003.
Waters B, Juels A, Halderman JA, Felten EW. New client puzzle outsourcing techniques for DoS resistance. In: The 11th ACM conference on computer and communications security (CCS) 2004.
Xu Y, Wang W. Detecting and mitigating DoS attacks in wireless networks without affecting the normal behaving nodes. In: IEEE military communication conference (Milcom'07) 2007.

Celia Li received her M. A. Sc. degree in Electrical & Computer Engineering from Ryerson University (Toronto, Canada) in 2005. She is currently a Ph.D. student at York University (Toronto, Canada), Department of Computer Science and Engineering. Her research interests include wireless networking, network security, and role-based access control.

Dr. Uyen Trang Nguyen received her Bachelor of Computer Science and Master of Computer Science degrees in 1993 and 1997, respectively, from Concordia University, Montreal, Canada. She completed her Ph.D. degree at the University of Toronto, Canada, in 2003. From 1995 to 1997 she was a software engineer at Nortel Networks, Montreal, Canada. She joined the Department of Computer Science and Engineering at York University, Toronto, Canada, in 2002 and is currently an Associate Professor. Her research interests are in the areas of mobile and ubiquitous computing, wireless networking, multimedia applications and network security.

Dr. Hoang Lan Nguyen received his B.E. degree with the highest honors in Telecommunications from the University of Wollongong (NSW, Australia) in 2003, and his M.Sc. and Ph.D. degrees in Computer Science from York University (Toronto, Canada) in 2006 and 2012, respectively. Before joining York University, he worked for Australia Nortel Networks as a software engineer. He is currently a research scientist at IBM Canada. His research interests include wireless networking, multicast routing, and network security.

Dr. Nurul Huda received his Ph.D. degree in Informatics in 2007 from the Graduate University for Advanced Studies (Tokyo, Japan). He was a post-doctoral fellow at the National Institute of Informatics (Tokyo, Japan) from 2007 to 2010, and a researcher at the Research Organization of Information and Systems (Tokyo, Japan) from 2010 to 2012. He is currently a research associate at York University (Toronto, Canada), Department of Computer Science and Engineering. His research interests are in the areas of wireless ad hoc networks, delay tolerant networks and privacy enhancing technologies.